[Extract Translation of Japanese Patent Publication No.H11-317783]

[Title] A header processing apparatus and a method thereof

[Abstract]

[Object] A pipeline processing method is provided. The pipeline processing method uses a plurality of general CPUs or a sequencer for the rapid processing of a packet header .

[Construction] A header processing apparatus and method comprises: first means 101 for extracting a field data of a protocol header from a part of a packet including a header divided from the packet, and inputting the extracted field data to an assigned memory; second means 102 for detecting data from the assigned memory using a detection key, and outputting a content of a table entry corresponding to the detection key; and third means 103 for preparing attribute information about a destination of the packet and about header update information, based on the an output from the second means and the data of the memory.

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平11－317783

(43)公開日　平成11年(1999)11月16日

(51)Int.Cl.[6]		識別記号	ＦＩ		
Ｈ０４Ｌ	29/02		Ｈ０４Ｌ	13/00	３０１Ｚ
	12/28			11/00	３１０Ｄ
	12/56			11/20	１０２Ａ

審査請求　有　　請求項の数17　ＯＬ　（全 18 頁）

(21)出願番号　特願平11－16006

(22)出願日　平成11年(1999)１月25日

(31)優先権主張番号　特願平10－11437

(32)優先日　平10(1998)１月23日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号

(72)発明者　村瀬　勉
東京都港区芝五丁目７番１号　日本電気株式会社内

(72)発明者　小林　正好
東京都港区芝五丁目７番１号　日本電気株式会社内

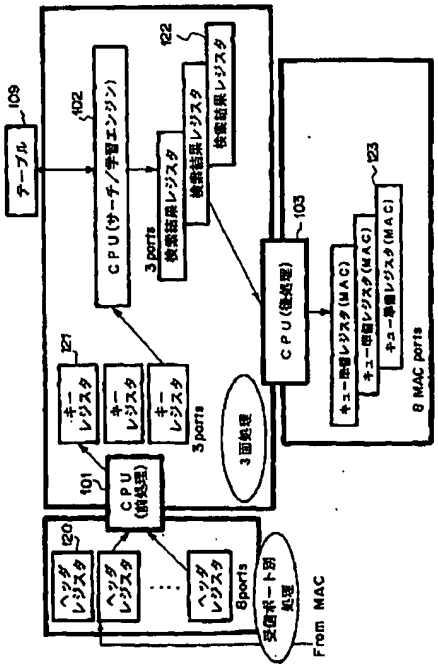
(74)代理人　弁理士　山下　穣平

(54)【発明の名称】　ヘッダ処理装置とそのヘッダ処理方法

(57)【要約】

【課題】　複数の汎用ＣＰＵやシーケンサを用い、パケットヘッダ処理の高速化のためパイプライン処理等の方法を提供する。

【解決手段】　ヘッダ処理装置及び該方法において、パケットから抽出されたヘッダを含むパケットの一部分から、プロトコルヘッダの各フィールドのデータを抽出し、指定された記憶場所に入力する第１の手段１０１と、該指定された記憶場所のデータを検索キーにして検索を行い、検索キーに一致するテーブルエントリの内容を出力する第２の手段１０２と、該第２の手段の出力内容と前記記憶場所のデータとから、前記パケットの送出先とヘッダ更新情報の属性情報を作成する第３の手段１０３と、を有することを特徴とする。



1

【特許請求の範囲】

【請求項１】　パケットから抽出されたヘッダを含むパケットの一部分から、プロトコルヘッダの各フィールドのデータを抽出し、指定された記憶場所に入力する第１の手段と、

該指定された記憶場所のデータを検索キーにして検索を行い、該検索キーに一致するテーブルエントリの内容を出力する第２の手段と、

該第２の手段の出力内容と前記記憶場所のデータとから、前記パケットの送出先とヘッダ更新情報の属性情報を作成する第３の手段と、を有することを特徴とするヘッダ処理装置。

【請求項２】　請求項１に記載のヘッダ処理装置において、前記第１～第３の手段は、１つのシーケンサと２つのＣＰＵ、２つのシーケンサと１つのＣＰＵ、３つのシーケンサ、３つのＣＰＵ、のいずれかにより構成されることを特徴とするヘッダ処理装置。

【請求項３】　請求項１に記載のヘッダ処理装置において、前記第１～第３の手段は、共有されるシーケンサと一つのシーケンサあるいは一つのＣＰＵ、共有されるＣＰＵと一つのシーケンサあるいは一つのＣＰＵ、のいずれかにより構成されることを特徴とするヘッダ処理装置。

【請求項４】　請求項１～３のいずれか１項に記載のヘッダ処理装置において、前記第１～第３の手段をそれぞれパイプライン処理する手段を有することを特徴とするヘッダ処理装置。

【請求項５】　請求項４に記載のヘッダ処理装置において、前記パイプライン処理について前記第１～第３の手段のいずれかで制御の受け渡しをハンドシェイクを用いて行うことを特徴とするヘッダ処理装置。

【請求項６】　請求項１～５のいずれか１項に記載のヘッダ処理装置において、前記第１～第３の各手段は、それぞれあらかじめ決められた固定時間の始まりあるいは終わりに、現固定時間内に、あるいは次の固定時間内にどの処理プログラムを実行するかを、決定することを特徴とするヘッダ処理装置。

【請求項７】　請求項１～５のいずれか１項に記載のヘッダ処理装置において、前記第１～第３の各手段は、それぞれあらかじめ決められ命令数の整数倍で処理を終了し、処理終了時あるいは開始時に、他の手段の処理の終了あるいは開始を確認し、少なくとも他の１つの手段において終了あるいは開始が確認できない場合には、命令の実行を一時停止し、全手段が処理の終了あるいは開始状態になるのを待つことを特徴とするヘッダ処理装置。

【請求項８】　請求項１～５のいずれか１項に記載のヘッダ処理装置において、前記第１の手段における記憶場所として、１つの共用部分と２つの個別部分を有し、該共用部分は、一部あるいは全てのフィールドデータを前記第２の手段と前記第３の手段で共用して使用するような記憶場所として用い、前記個別部分は、各個別部分に同一内容のデータを持ち且つ各個別部分のそれぞれを、前記第２の手段と、前記第３の手段で独立に用いることを特徴とするヘッダ処理装置。

2

【請求項９】　請求項２～５のいずれか１項に記載のヘッダ処理装置において、前記第１の手段におけるシーケンサの命令として、ヘッダ可能性部分から、ＤＩＸ規格とＩＥＥＥ規格８０２．３のＭＡＣ（Media Access Control）アドレスと、８０２．１Ｑと、ＩＰと、ＴＣＰと、ＵＤＰと等の固定プロトコルフィールドを抽出する特殊演算からなる第１の命令セットと、任意の複数ビットを抽出する基本演算からなる第２の命令セットとを持ち、前記第１の命令セットと前記第２の命令セットの命令を各々固有のハードウエアで処理することを特徴とするヘッダ処理装置。

【請求項１０】　請求項２～５のいずれか１項に記載のヘッダ処理装置において、前記第２の手段におけるシーケンサの命令として、テーブルへの検索データ入力と前記テーブルからの検索結果のレジスタへの蓄積を行う特殊演算からなる第１の命令セットと、任意の複数ビットをレジスタからレジスタへ移動する基本演算からなる第２の命令セットとを持ち、前記第１の命令セットと前記第２の命令セットの命令を各々固有のハードウエアで処理することを特徴とするヘッダ処理装置。

【請求項１１】　請求項２～５のいずれか１項に記載のヘッダ処理装置において、前記第３の手段におけるシーケンサの命令として、検索結果出力から装置内パケットヘッダの作成を行なう特殊演算からなる第１の命令セットと、任意の複数ビットをレジスタからレジスタへ移動する基本演算からなる第２の命令セットとを持ち、前記第１の命令セットと前記第２の命令セットの命令を各々固有のハードウエアで処理することを特徴とするヘッダ処理装置。

【請求項１２】　パケットから抽出されたヘッダを含むパケットの一部分から、プロトコルヘッダの各フィールドのデータを抽出し、指定された記憶場所に入力する第１の手順と、該指定された記憶場所のデータを検索キーにして検索を行い、前記検索キーに一致するテーブルエントリの内容を出力する第２の手順と、該第２の手順の出力内容と前記記憶場所のデータとから、前記パケットの送出先とヘッダ更新情報の属性情報を作成する第３の手順と、を有することを特徴とするヘッダ処理方法。

【請求項１３】　請求項１２に記載のヘッダ処理方法において、前記第１～第３の手順は、１つのシーケンサと２つのＣＰＵ、２つのシーケンサと１つのＣＰＵ、３つのシーケンサ、３つのＣＰＵ、のいずれかを用いて実行されることを特徴とするヘッダ処理方法。

【請求項１４】　請求項１２に記載のヘッダ処理方法において、前記第１～第３の手順は、共有されるシーケンサと一つのシーケンサあるいは一つのＣＰＵ、共有され

るＣＰＵと一つのシーケンサあるいは一つのＣＰＵ、のいずれかを用いて実行されることを特徴とするヘッダ処理方法。

【請求項１５】　請求項１２～１４のいずれか１項に記載のヘッダ処理方法において、前記第１～第３の手順をそれぞれパイプライン処理によって行うことを特徴とするヘッダ処理方法。

【請求項１６】　請求項１５に記載のヘッダ処理方法において、前記パイプライン処理について前記第１～第３の手順のいずれかで制御の受け渡しをハンドシェイクを用いて行うことを特徴とするヘッダ処理方法。

【請求項１７】　請求項１２～１６のいずれか１項に記載のヘッダ処理方法において、前記第１～第３の各手順は、それぞれあらかじめ決められた固定時間の始まりあるいは終わりに、現固定時間内に、あるいは次の固定時間内にどの処理プログラムを実行するかを、決定することを特徴とするヘッダ処理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、パケットのヘッダ処理装置及びヘッダ処理方法に関し、特にパケット転送用手段を有してパケットを高速に処理するヘッダ処理装置及びヘッダ処理方法に関する。

【０００２】

【従来の技術】従来、公衆電話回線はアナログの音声通信であったが、情報化時代に入ってからデータの通信が盛んになり、今後益々データの交錯状態が増加し、更に大量のデータを高速に送信するために、パケット形式で、伝送されることが多くなっている。

【０００３】このパケット形式によるヘッダ処理は、パケットの宛先のある出力インタフェースを求めることと、該出力インタフェースに適するヘッダに変換するための情報を求めることにある。これらの作業は、ヘッダの各種フィールドなどとフォーワーディングテーブルを用いて行うもので、非常に多数の処理ステップを必要とする。例えば、マルチレイヤスイッチ（これは物理レイヤの上位レイヤであるレイヤーＬ２とＬ３の処理を、１つの装置あるいはＬＳＩチップにおいて行うスイッチである。）においては、Ｌ２のパケット（ＭＡＣパケット）のヘッダからＭＡＣ（媒体アクセス制御：Media Access Control）のアドレスを取り出し、下記の４種類のＭＡＣフォーマットのどちらであるかを識別し、識別結果に応じて決まるヘッダ位置から、Ｌ３のパケット（ＩＰパケット）を取り出し、ＩＰパケットのヘッダからＩＰのアドレスを取り出す。

【０００４】さらに、ＭＡＣアドレスが所定のアドレスであるかどうかをテスト（ルータテスト）し、そうでないときには、Ｌ２の処理を行い、所定のアドレスであるときにはＬ３の処理を行う。前記識別結果によっては、Ｌ３のパケットを取り出す処理を行わず、Ｌ２処理を行う。ルータテストについて、Ｌ２及びＬ３の処理では、フォワーディングテーブルを検索するという処理を行う。このフォワーディングテーブルは一般にかなり大きいため、この検索処理は非常に時間のかかるものとなる。

【０００５】４種類のＭＡＣフォーマットとは、図１４に示すように、Ethernet IIとも呼ばれるＤＩＸ規定とＩＥＥＥ８０２．３規定という２つの標準形式と、それぞれに８０２．１Ｑ規定とも呼ばれる仮想ＬＡＮの識別子がついているかいないかの２種類との積で４通りである。例えば、ＤＩＸ規定と８０２．１Ｑ規定との組合せでは、ＩＰ　ＰＤＵとＶＬＡＮ　ＩＤとがＤＩＸ規定の単独に対して余分に設定されている。また、ＬＡＮエミュレーションの標準化の折、そのサービスは既存のイーサネット及びトークンリンクのＬＡＮのアプリケーションをエミュレートするもので、相互接続・運用の際にＭＡＣレベルでの接続が必須である。さらに、このＭＡＣパケットのペイロードとして、ＩＰパケットがあり、さらにそのＩＰパケットのペイロードとして、ＴＣＰパケット、ＵＤＰパケットのいずれかがある。

【０００６】このようにパケットヘッダは、それを順に解析して始めて、内容フォーマットが明らかになるものであるため、ヘッダ解析前のパケット受信時には、パケットヘッダがどこまでなのかは不明である。しかし、取り扱うプロトコルは定まっているため、最大のヘッダを想定して、この部分つまり、ヘッダを必ず含むようなパケットの一部分（ヘッダ可能性部分と呼ぶ）をパケットから取り出しておけば、不要な情報が含まれている可能性はあるが、必要な情報を取りこぼすことはない。さらに、検索結果を適当に加工して、パケットを宛先に送出する準備を行う必要がある。この準備には、パケットを宛先行きの待ち行列に加える、パケットヘッダを書き換える、パケットをコピーするなどの処理がある。

【０００７】以上のようなヘッダ処理を１つのＣＰＵで順に行うことで、１つのパケットの処理が終了する。高速化を図るためには、この処理を高速に行う必要がある。

【０００８】また、汎用のＣＰＵを用いた場合には、論理回路で構成したハードウエアでは、１ステップ、つまり１クロックで実行できる機能を、２クロック以上必要とすることがある。

【０００９】例えば、ハードウエア論理回路において、メモリＡとメモリＢの内容を２倍したものの加算を行い、メモリＣに保存する（Ｃ＝２×Ａ＋２×Ｂ）には、１クロック目でメモリＡ，Ｂを同時に読み出してレジスタに入れ、２クロック目でそのレジスタを１ビットシフトしながら（２倍化）、加算し、その結果をメモリＣに書き込むことができる（特殊演算と呼ぶ）。

【００１０】一方、汎用ＣＰＵの場合には、加算やシフトといった最も基本的な演算（基本演算と呼ぶ）を用い

て、メモリＡをレジスタに読み出し（１クロック目）、メモリＢをレジスタに読み出し（２クロック目）、メモリＡのレジスタを１ビットシフトして（３クロック目）、メモリＢのレジスタを１ビットシフトして（４クロック目）、メモリＡのレジスタとＢのレジスタを加算して（５クロック目）、結果をメモリＣに書き込む（６クロック目）ことになる。

【００１１】この例で示すように、前者は２クロックで、後者は６クロックを要し、論理回路で構成したハードウエアは、汎用ＣＰＵよりも高速に処理ができる。ただし、ハードウエア論理回路は、修正変更や将来の機能追加が困難である。そのため、ハードウエア論理回路は設計変更の柔軟性に欠ける。従来、ハードウエア論理回路におけるこの柔軟性を欠くという欠点を改善するためにシーケンサというプログラマブルな専用ＣＰＵを用いる方法があり、信号処理などの分野で使用されていた。

【００１２】シーケンサも汎用ＣＰＵも、一般に、ＲＯＭ／ＲＡＭに入れられたプログラムを持ち、プログラムカウンタと呼ばれるポインタが指し示すアドレスの内容（命令）が取り出され、命令が解読（デコード）され、デコード信号が出され、デコード信号を受けた演算回路が動作するというものである。

【００１３】しかしながら、シーケンサでは、演算回路が専用に作られており、例えば、レジスタＡとレジスタＢの内容をそれぞれ２倍して加算するという演算を、１クロックで行なうような演算を組み込むことができる。従って、汎用ＣＰＵよりも高速に演算が行われる。信号処理のシーケンサでは、Ｘ，Ｙを変数とし、Ａ，Ｂを媒介変数とする、Ｙ＝ＡＸ＋Ｂ等の一次関数の演算が演算回路として組まれている。しかしながら、パケットのヘッダを処理するためのシーケンサはなかった。

【００１４】

【発明が解決しようとする課題】上記従来技術の第１の問題点は、ヘッダ処理において、順序よく行うべきことと、同時並行的に（独立に）行うことが可能ではあるが、汎用ＣＰＵにおいては、すべて順に行われていたことにある。従って、高速化を図るには汎用ＣＰＵの能力を上げる必要があるが、能力を上げることは、汎用ＣＰＵそのものの性能アップのため、コスト高につながる。

【００１５】また、汎用ＣＰＵを多数用意し、パイプライン処理を行う場合には、適当なパイプライン数に設計しなければ、やはりコスト高になる。あるいはフォワーディングテーブルのような資源がボトルネックになり、汎用ＣＰＵが能力を十分活かせなくなる恐れがある。

【００１６】また、第２の問題点は、汎用ＣＰＵのかわりにハードウエア論理回路を用いた場合には、設計変更の柔軟性に欠けるという問題がある。これに対し、シーケンサというプログラマブルな専用ＣＰＵを用いる方法があり、信号処理などの分野で使用されていた。しかしながら、パケットのヘッダ処理を高速に行なうためのこ

のようなシーケンサは、いまだ公開されていない。

【００１７】［発明の目的］本発明は、ボトルネックになるフォワーディングテーブルのような資源で制約される速度まで、パケット等のヘッダ処理の高速化を汎用ＣＰＵを用いて低コストで行うことを目的とする。また、パケットヘッダ処理をプログラマブルなハードウエア論理回路、即ちシーケンス処理で実現することで、高速性を保ちながら、設計変更や将来の機能追加を低コストで実現することを目的とする。

【００１８】

【課題を解決するための手段】上記目的を達成するために、本発明は、パケットから抽出されたヘッダを必ず含むようなパケットの一部分から、プロトコルヘッダの各フィールドのデータを抽出し、指定された記憶場所に入力する第１の手段と、該指定された記憶場所のデータを検索キーにして検索を行い、検索キーに一致するテーブルエントリの内容を出力する第２の手段と、該出力内容と該記憶場所のデータとから、パケットの送出先やヘッダ更新情報といった属性情報を作成する第３の手段とを持つことを特徴とするヘッダ処理装置を提供するものである。

【００１９】また本発明はパケットから抽出されたヘッダを含むパケットの一部分から、プロトコルヘッダの各フィールドのデータを抽出し、指定された記憶場所に入力する第１の手順と、該指定された記憶場所のデータを検索キーにして検索を行い、前記検索キーに一致するテーブルエントリの内容を出力する第２の手順と、該第２の手順の出力内容と前記記憶場所のデータとから、前記パケットの送出先とヘッダ更新情報の属性情報を作成する第３の手順と、を有することを特徴とするヘッダ処理方法を提供するものである。

【００２０】［作用］パケットのヘッダ処理には、処理時間に注目すると主に以下の３つの処理が大半を占める。すなわち、第１の処理として、パケットのヘッダの構造を明らかにし、ヘッダの書くフィールドを抽出すること、第２の処理として、アドレスフィールドの情報、つまりアドレスからパケットの行き先を決定すること、第３の処理として、パケットの行き先へパケットを送出するのに必要なヘッダ変換を行うことである。

【００２１】かかる処理を行うＣＰＵの構成概念図を図５に示す。ＣＰＵによる場合、ＣＰＵ１０内には加算器（＋）と乗算器（×）とレジスタＲ１，Ｒ２，Ｒ３と命令デコーダとから成り、バスラインを介してＣＰＵ１０の基本ソフトのＯＳ１１とプログラム１２とから構成される。かかる構成の場合、プログラム１２の内容の例によれば、Ａ～Ｄの数値と各レジスタとから、３Ａ＋３Ｂ＝Ｄを得るのに９ステップを要しているが、コストパフォーマンスとしては申し分ない。

【００２２】また、専用のシーケンスによる構成概念図を図２に示す。専用のシーケンスとしては、命令デコー

ダを有するシーケンサ２０と、機能処理Ｆ１（２１），Ｆ２（２２）と、シーケンサ２０が動作する手順を格納するプログラム２３とから構成される。かかる構成の場合、専用の機能処理手段を有しているので、図５と同様な結果を得るために、２ステップでよく、高速化のために優れている。

【００２３】さらに、本発明により、パケットヘッダの一連の第１～第３の処理をパイプライン的に行うために、順に行うことが必要な前記３つの部分に分け、なおかつボトルネックになる資源の使用を最小限にとどめ、なおかつ該資源を使用するＣＰＵが１００％稼動するような分割とし、各部分では、それぞれのＣＰＵを割り当てて独立に処理を行うことができる。これにより、最適なＣＰＵ数が選択でき、最も負荷がかかるＣＰＵを１００％あるいは１００％近くまで活用することができる。

【００２４】また、高速にパケットヘッダの処理を行うにはこれらを高速に行う必要がある。従来は、１つのＣＰＵが上記３処理を順に行っていたが、ハードウエアで同様のことを行う場合、３処理を別々に行うことで、パイプライン処理が可能となる。

【００２５】かかるパイプライン的な構成による処理の概念的な例を図３に示す。処理Ａと処理Ｂ，処理Ｃとは同一の固定時間で処理が終わり、スタートしてから終了するまでも固定時間に設定されており、一連の処理が必要な場合、同一の固定時間で処理可能な処理Ａと処理Ｂ，処理Ｃとがあり、ＣＰＵ１で処理Ａが終了してからＣＰＵ２で処理Ｂが開始されるとともにＣＰＵ１で次の処理Ａが開始される、ＣＰＵ２で処理Ｂが終了してからＣＰＵ３で処理Ｃが開始されるとともにＣＰＵ２で次の処理Ｂが開始されるような処理をパイプライン的な処理と称している。

【００２６】また、処理Ａ～Ｃが同一の固定時間で処理が終わらない場合に、プログラム処理する際には、種々な方式があり、その３例を図４に示して説明する。ここでは、ＣＰＵをシーケンサに置き換えて説明する。図４（ａ）は各処理の時間が異なる場合、処理プログラムＡ４１が処理プログラムＢ４２より短いとき、処理プログラムＢ４２が終了するまでレジスタ４３の処理プログラムＢ４２に対応するフラグが立たないので、処理プログラムＡ４１を処理するシーケンサ１は待ち受け状態とし、処理プログラムＢ４２が終了してレジスタ４３のフラグが立って、シーケンサ１は続く処理プログラムＡを処理する。また、図４（ｂ）では、処理プログラムＡ４１が処理プログラムＢ４２より短いとき、シーケンサ１は処理プログラムＡ４１が終了してから所定の一定時間を待って、レジスタ４３の処理プログラムＢ４２に対応するフラグが立っているのかどうかを検出し、フラグが立った場合は次の処理プログラムＡ４１を処理する。一方、フラッグが立っていない場合は所定の一定時間を待ち、再度、レジスタ４３の処理プログラムＢ４２に対応するフラグが立っているのかどうかを検出する。即ち、待ち受け状態に一定時間を設定しておくことで、シーケンサ１の負担を軽減している。さらに、図４（ｃ）では、処理プログラムＡ４１が処理プログラムＢ４２より短いとき、所定の一定クロック数毎に、シーケンサ１がレジスタ４３の処理プログラムＢ４２に対応するフラグが立っているのかどうかを検出するもので、所定の一定クロック数の複数倍毎にフラグを検出することで、更にシーケンサの負担を軽減することができる。

【００２７】

【発明の実施の形態】次に、本発明の実施の形態について図面を参照して詳細に説明する。

【００２８】＜第１の実施形態＞本発明の第１の実施形態は、図１に示すように、第１～第３の手段となる３つの汎用ＣＰＵと各種レジスタから構成される。ＣＰＵ（前処理）と名づけられた第１の汎用ＣＰＵ１０１は、ヘッダレジスタ群（単にヘッダレジスタと称する）１２０の情報から、検索キーレジスタ群（単に検索キーレジスタと呼ぶ）１２１の内容を生成する。

【００２９】ＣＰＵ（検索（サーチ）／学習エンジン）と名づけられた第２の汎用ＣＰＵ１０２は、検索キーレジスタ１２１の情報からフォワーディングテーブル１０９を検索し、検索結果により検索結果レジスタ群（単に検索結果レジスタと呼ぶ）１２２を生成する。

【００３０】さらに、ＣＰＵ（後処理）と名づけられた第３の汎用ＣＰＵ１０３は、検索結果レジスタ１２２の情報からキューバッファ準備レジスタ群（単にキューバッファ準備レジスタと呼ぶ）１２３の内容を生成する。

【００３１】ヘッダレジスタ１２０とキューバッファ準備レジスタ１２３は、受信インタフェース数、すなわち同時にパケットが到着する数Ｎだけ必要である。検索結果レジスタ１２２は、１つしかないフォワーディングテーブル１０９の検索に関係するので、Ｎ以下でよいが、任意の１つのレジスタのアクセスを２つの汎用ＣＰＵが同時に行わないという制約のもとで動作させるため、３つ必要である。

【００３２】その３つの検索結果レジスタ１２２は、それぞれ、前処理汎用ＣＰＵ１０１の書き込み、サーチ／学習エンジン１０２の書き込み、後処理汎用ＣＰＵ１０３の読み取りに同時に使用される。この検索結果レジスタ１２２と対応させて制御させるため、検索キーレジスタ１２１も３つ用意する。ただし、検索キーレジスタ１２１においては、その２つは、それぞれ、前処理汎用ＣＰＵ１０１の書き込み、検索エンジン１０２の読み取りに同時に使用され、１つは休止となる。

【００３３】以下、３つの汎用ＣＰＵについて説明する。パケットあたりの処理において、検索／学習エンジン１０２がフォワーディングテーブル１０９の検索に時間がかかるため、もっともボトルネックとなる。また、この時間をできるだけ短縮するために、検索回数を最小

限にするための処理を前処理汎用ＣＰＵ１０１と後処理汎用ＣＰＵ１０３に任せる。

【００３４】前処理と後処理のそれぞれの汎用ＣＰＵ１０１，１０３の処理を説明する。前処理汎用ＣＰＵ１０１は、ヘッダ解析を行い、ＭＡＣパケットのヘッダフォーマットの決定および必要に応じて、ＭＡＣパケットのペイロード、つまりＩＰパケットのヘッダ解析を行う。ＭＡＣパケットのペイロードは必ずしもＩＰパケットではない。これは、ＭＡＣアドレスがルータのＭＡＣアドレスであれば、該パケットはＩＰパケットとしてレイヤーＬ３を処理し、そうでなければＩＰパケットであるかどうかを調べる必要はなく、レイヤーＬ２の処理を行う。従来の技術では、ＭＡＣアドレスがルータ宛てのＭＡＣアドレスかどうかは、フォワーディングテーブル１０９を検索していた。しかし、これは検索エンジン１０２の負荷になるので、この前処理汎用ＣＰＵ１０１にて、その判定を行う。

【００３５】また、判定でルータ宛先の解析が必要で、つまりＬ３の処理を行う必要がある場合には、ＩＰパケットのヘッダの解析に入る。ＩＰヘッダについては、宛先ＩＰアドレスを抽出し、それを検索キーとして検索キーレジスタ１２１に入力する。また、受信ポート番号やＶＬＡＮ－ＩＤといった情報は、検索キーレジスタ１２１に書き込むとともに検索結果レジスタ１２２にも書き込む。これらの情報は、キューバッファ準備レジスタ１２３への書き込み時にも必要だからである。

【００３６】後処理汎用ＣＰＵ１０３は、検索結果レジスタ１２２の受信ポート番号と一致する番号のレジスタを指定するとともに、検索エンジン１０２により検索結果レジスタ１２２に入力された検索結果をキュー準備レジスタ１２３において指定されたフォーマットに変換する。例えば、宛先がマルチキャストである場合には、検索エンジン１０２は、フォワーディングテーブル１０９から、ビットマップを引き出してきて、それを検索結果レジスタ１２２に入力するので、それを後処理汎用ＣＰＵ１０３が、そのビットマップを解析し、ビットのｏｎ／ｏｆｆをそのビットに対応する出力ポートへのパケット送出という命令列に変換する。

【００３７】検索／学習エンジン１０２は、検索キーレジスタ１２１に入力された検索キーと、Ｌ２であるかＬ３であるかどちらの処理を行うべきかを示す情報を用いて、それに対応するフォワーディングテーブル１０９を検索する。検索により、キーと一致するものが発見された場合には、そのテーブル内で発見されたキーに結合されている情報を引き出す。該情報としては、出力ポート番号、出力時に変更すべきアドレスフィールド名と、変更に用いるアドレス、などが含まれている。

【００３８】以上のように３つの汎用ＣＰＵにより独立に処理を行うことで、一つの汎用ＣＰＵで処理を行う場合に問題になる高速化の限界を超えることができる。また、汎用ＣＰＵを４つ以上使用する必要がなく、３つで十分に処理を分散することができる。

【００３９】なお、上記実施形態では、処理の一部は前処理を経過して行う処理が必要であるが、重複して処理を行うことができるので、図５に示したＣＰＵによりシーケンス的にパケット処理の高速化が達成できる。

【００４０】＜第２の実施形態＞本発明の第２の実施形態は、第１の実施形態における各汎用ＣＰＵの代わりにシーケンサを用いる構成である。これにより、汎用ＣＰＵを用いた場合よりもさらに高速化できる。図６にシーケンサによる処理の例を示している。シーケンサの動作及びシーケンサの構成は、図２に示したものであり、本実施形態による動作は、図１に示した第１の実施形態と同様なので、説明を省略する。なお、処理ステップが機能処理により極めて短時間で処理することが可能である。

【００４１】＜第３の実施形態＞本発明の第３の実施形態は、第１あるいは第２の実施形態において、各ＣＰＵ、又は各シーケンサがパイプラインとして動作し、以下のような構造を持つものである。図７は各シーケンサがパイプラインとして動作する場合を示す図である。

【００４２】各シーケンサ１０１，１０２，１０３は、固定時間でその処理を終わるように固定時間（周期）を決めてある。システムを動作開始させるときに、同一のタイミングで信号が出るような周期そろえクロックを周期タイマ２０１から、シーケンサ１０１，１０２，１０３に同時に入力することで、周期の始まりを同一にする。これにより各シーケンサにおける処理は、次の手段のことを考慮せずに独立に動作することが可能となる。図７において、シーケンサに代えて汎用ＣＰＵを用いることも可能である。

【００４３】これにより、汎用ＣＰＵあるいはシーケンサ制御において、汎用ＣＰＵ間の連絡に必要な手続きが削除でき、或いは各シーケンサ間の連絡に必要な手続きが削除できるので、その分を本来の処理に向けることができ、さらなる高速化がはかれる。

【００４４】＜第４の実施形態＞本発明の第４の実施形態は、図６の第２の実施形態において、図８に示すように、以下のような構造を持つものである。

【００４５】各レジスタ群１２０～１２３は、それぞれ処理を行なうために必要なデータ、すなわち処理の入力データがあるかどうかを示す第１のレディレジスタをそれぞれ持つ。即ち、各レジスタ群１２０～１２３の各レジスタ１２０ａ～１２０ｃ，１２１ａ～１２１ｃ，１２２ａ～１２２ｃ，１２３ａ～１２３ｃはそれぞれレディレジスタ３０１，３０２，３０３を持っている。さらに、各レジスタ群１２０～１２３は、処理の結果を出力するための記憶場所が使用できるかどうか、すなわち前パケットのヘッダ処理の結果が、次の手段で入力として使用済みかどうかを示す第２のレディレジスタをそれぞ

れ持つ。即ち、各レジスタ群１２０～１２３の各レジスタ１２０ａ～１２０ｃ，１２１ａ～１２１ｃ，１２２ａ～１２２ｃ，１２３ａ～１２３ｃはそれぞれレディレジスタ４０１，４０２，４０３を持っている。

【００４６】以下、上記構造の動作について説明する。

【００４７】例えば、シーケンサ１０２は、処理の最初において、読み出し側レジスタ群（検索キーレジスタ群）１２１の第１のレジスタ１２１ａの第１のレディレジスタ３０１と書き込み側レジスタ群（検索結果レジスタ群）１２２のレジスタ１２２ａの第２のレディレジスタ４０１を常に読み出し、第１のレディレジスタ３０１が「入力データあり」の場合で、かつ、第２のレディレジスタ４０１が「書き込みＯＫ」の場合にのみ、読み出し側レジスタ群１２１のレジスタ１２１ａからデータを読み、データ処理して書き込み側レジスタ群１２２のレジスタ１２２ａにデータを書き込む。その後、読み出し側レジスタ群１２１のレジスタ１２１ａの第２のレディレジスタ４０１に「書き込みＯＫ」をセットし、書き込み側レジスタ群１２２のレジスタ１２２ａの第１のレディレジスタ３０１に「入力データあり」をセットする。さらに、読み出し側レジスタ群１２１のレジスタ１２１ａの第１のレディレジスタ３０１に「入力データなし」をセットし、書き込み側レジスタ群１２２のレジスタ１２２ａの第２のレディレジスタ４０１に「書き込み禁止」をセットする。

【００４８】一方、読み出し側レジスタ群１２１のレジスタ１２１ａの第１のレディレジスタ３０１が「入力データなし」の場合、あるいは書き込み側レジスタ群１２２のレジスタ１２２ａの第２のレディレジスタ４０１が「書き込み禁止」の場合には、処理を開始せず、第１のレディレジスタ３０１あるいは第２のレディレジスタ４０１を読み出すだけのウエイト状態を続ける。

【００４９】２つのシーケンサは少なくとも３つのレジスタを有し、これらの３つのレジスタは２つのシーケンサによって排他的に用いられる。例えば、シーケンサ１０１が検索キーレジスタ群１２１の１つのレジスタ１２１ａのデータ書き込みを行っている場合、シーケンサ１０２は検索キーレジスタ群１２１の他の１つのレジスタ１２１ｂからデータの読み込みを行い、検索キーレジスタ群１２１の残りの１つのレジスタ１２１ｃは休止となる。

【００５０】上記排他的動作について、各レジスタ群と各シーケンサの動作を示す図８を用いて、さらに説明する。

【００５１】複数のシーケンサはあるレジスタを用いてデータを受け渡すのであるが、レジスタへの書き込みと読み出しとが同時に起こると、データの妥当性が無くなる恐れがあるために、書き込みと読み出しを排他的に制御することが求められる。以下、排他的な動作について説明する。

【００５２】例えば、検索キーレジスタ群１２１の動作に着目して説明すると、検索キーレジスタ群１２１の第１のレジスタ１２１ａがシーケンサ１０１の書き込みに使用されているときには、検索キーレジスタ群１２１の第２のレジスタ１２１ｂは、シーケンサ１０２の読み出しに使用されており、検索キーレジスタ群１２１の第３のレジスタ１２１ｃはどこからもアクセスされない状態、またはシーケンサ１０１，１０２のどちらか一方の使用が可能な状態である。

【００５３】シーケンサ１０１，１０２は、各々の処理を終了したら、次の処理のために別のレジスタを使用する。つまり、検索キーレジスタ群１２１の第２のレジスタ１２１ｂはシーケンサ１０１の書き込みに使用され、検索キーレジスタ群１２１の第３のレジスタ１２１ｃは、シーケンサ１０２の読み出しに使用されており、検索キーレジスタ群１２１の第１のレジスタ１２１ａはどこからもアクセスされない状態、またはシーケンサ１０１，１０２のどちらか一方の使用が可能な状態となる。

【００５４】このような動作を行うことで、各レジスタは読み込みと書き込みが排他的に行われるため、レジスタへの書き込みと読み出しが同時に起こることに起因するデータの妥当性の心配は解消される。

【００５５】以上説明したように、本実施形態では、ハンドシェイクを用いた確実なパイプライン制御が行なえ、あるいは各手段で周期が異なることも可能であるし、周期の始まりを一致させる必要もなくなり、パイプライン制御の場合には、もっとも周期の長いものに全手段を合わせる必要があるのに比べて、より短時間で処理を終了することができる。なお、図８において、シーケンサに代えて汎用ＣＰＵを用いて第１の実施形態の構成に用いることも可能である。

【００５６】＜第５の実施形態＞上記本発明の第４の実施形態においては、シーケンサ１０１～１０３の各シーケンサは、第１のレディレジスタが「入力データなし」、あるいは第２のレディレジスタが「書き込み禁止」の場合、常に読み出しを行うウエイト状態としていたが、本第５の実施形態では図４（ｂ）を用いて説明した動作と同様に、決められたウエイト時間の間、読み出しを行わないウエイト状態になる構成とした。これにより、制御がさらに簡単になるため、そのぶん処理を高速化できる。

【００５７】なお、本実施形態ではそれぞれあらかじめ決められた固定時間の始まりに現固定時間内にどの処理プログラムを実行するかを決定しても、固定時間の終わりに、次の固定時間内にどの処理プログラムを実行するかを、決定してもよい。

【００５８】＜第６の実施形態＞本発明の第６の実施形態は、第５の実施形態において、シーケンサの処理時間が一定ではないが、ある決められた基準周期の整数倍となる場合である。

【００５９】例えば、シーケンサ１０１とシーケンサ１０２との処理時間がともにある決められた基準周期の整数倍であるが、その処理時間が異なる場合に、シーケンサ１０１とシーケンサ１０２との処理時間の差も基準周期の整数倍となる。そのため、本実施形態では第５の実施形態における決められたウエイト時間を基準周期の整数倍とした。

【００６０】例えば、図４（ｃ）を用いて具体的に説明すると、例えば基準周期を１００クロックとした場合、シーケンサ１は処理プログラムＡの終了後、１００クロック毎にレジスタを読み出して、処理プログラムＢ用のレジスタのフラグの状態を検出する。また、１００クロック毎とせず、１００クロックの整数倍の４００クロック目にレジスタの状態を読み出し、フラグが立っていたならば、次の処理プログラムＡの処理を開始する。これにより、処理を開始する必要のない時刻が決まるため、制御が簡単になるため、そのぶん処理を高速化できる。

【００６１】＜第７の実施形態＞本発明の第７の実施形態は、図１０に示すように、第２の実施形態において、第１のシーケンサ１０１が処理結果を蓄積するレジスタとして、キーレジスタ１２１と検索結果レジスタ１２２を両方使用し、第２のシーケンサ１０２が処理結果を蓄積するレジスタとして、検索結果レジスタ１２２を使用し、第３のシーケンサ１０３が処理結果を蓄積するレジスタとして、キュー準備レジスタ１２３を使用する構成である。つまり、検索結果レジスタ１２２は、第１のシーケンサ１０１と第２のシーケンサ１０２の両方から書き込みがあるレジスタである。ただし、検索結果レジスタ１２２は、該両方のシーケンサ１０１，１０２から同時にアクセスされることはない。これは、第１の実施形態でも述べたように、レジスタ１２２は、３個用意されており、３つのシーケンサそれぞれが、１つを読むあるいは書くことに使用しているためである。

【００６２】第１のシーケンサ１０１は、第１のレジスタ１２１に検索に必要な情報を書き込み、同時に第２のレジスタ１２２に第３のシーケンサ１０３が必要とする情報（スルー情報）をパケットヘッダ可能部分から抽出して、書き込む。その他は、（ＣＰＵをシーケンサに書き換えた）第１の実施形態と同様である。

【００６３】これにより、（ＣＰＵをシーケンサに書き換えた）第１の実施形態においては必要である第２のシーケンサ１０２がスルー情報をレジスタ１２２からレジスタ１２３へ移動させるという処理が必要ではなくなることで、制御が簡単になるため、そのぶん処理を高速化できる。

【００６４】本実施形態で、シーケンサに代えて汎用ＣＰＵを用いる構成も可能である。また、シーケンサ１０１はサーチ／学習シーケンサ１０２が管理する検索結果レジスタ１２２に直接作用することを示したが、サーチ／学習シーケンサ１０２がキュー準備レジスタ１２３に直接書き込むことも可能である。

【００６５】＜第８の実施形態＞本発明の第８の実施形態は、図１１，図１２，図１３に示すように、第２～７の実施形態における各シーケンサの行なう処理において、該処理は複数の命令から構成されており、図１２に示すように、１つの命令の実行が加算やシフト算や、比較算のように汎用ＣＰＵにおいて用いられているような基本演算を用いて、あるいは１つの命令が、あるデータの加算した結果を比較しながら同時に別のデータをシフトするといったような該基本演算の並列あるいは順次的組み合わせで構成されている特殊演算を用いて実行される演算と命令で構成されている。これらの演算はどちらもハードウエアで構成されている。

【００６６】以下、汎用ＣＰＵとシーケンサの処理クロック数の比較について説明する。

【００６７】図１１に示すように、汎用ＣＰＵ６００においては、命令セット６０４から取り出された命令は、レジスタ６０３の情報を用いて、第１の基本演算６０１あるいは第２の基本演算６０２を用いて演算結果を出し、再度レジスタ６０３に書き込む。

【００６８】また、図１２に示すように、本実施形態のシーケンサ７００においては、命令セット７０４から取り出された命令は、レジスタ７０３の情報を用いて、特殊演算７０１あるいは基本演算６０２あるいは基本演算６０１を用いて演算結果を出し、再度レジスタ７０３に書き込む。

【００６９】例えば、ＸとＹという２つの情報があるレジスタにあり、この２つを用いて、Ｚ＝２×Ｘ＋２×Ｙを求め、Ｚをレジスタ Ｚに書き込むという処理を行なう。基本演算としては、図１２に示すように加算６０２及びシフト６０１を持っており、基本演算を用いて、命令を並べると以下のようになる。各命令は１クロックで実行する。レジスタＸを１ビットシフトして（１クロック目）、レジスタＹを１ビットシフトして（２クロック目）、レジスタＸとレジスタＹとを加算して（３クロック目）、結果をレジスタＺに書き込む（４クロック目）。

【００７０】一方、図１３に示すように特殊演算７０１は、ビットシフト８０１とビットシフト８０２と加算８０３とを持ち、Ｚ＝２×Ｘ＋２×Ｙを求め、Ｚをレジスタ Ｚ８１３に書き込むという処理を１クロックで行なう演算（演算Ｚと呼ぶ）である。従って、特殊演算を用いて命令を構成すると、演算Ｚ（１クロック目）だけとなる。このように、あらかじめ必要な演算を特殊命令として、ハードウエアで実現しているため、処理の高速化が行なえる。一方、基本演算を組み合わせることで、特殊演算を実現できるようにしておけば、ハードウエアでは汎用ＣＰＵよりも同じ処理を高速に実現できるため高速化が行なえ、また命令の組み合わせの変更で演算の組み合わせの変更が可能になるため、柔軟性も得られ、設計

15

変更や新規処理の追加が高速性という利点を保ちつつ、低コストで実現できる。

【００７１】上記各実施形態において、パケットのヘッダ処理装置及びヘッダ処理方法について説明したが、これらはＭＡＣヘッダやＴＣＰ／ＩＰヘッダに限らず、他のヘッダ処理にも適用できるものである。また、汎用ＣＰＵやシーケンサの数を３個として説明したが、もっと多数の汎用ＣＰＵ或いはシーケンサを用いることで更なる高速化を図ることができる。

【００７２】また、各実施形態において、第１～第３の手段をＣＰＵ１０１～１０３又はシーケンサ１０１～１０３で構成した例を示したが、第１～第３の手段は１つのシーケンサと２つのＣＰＵ、２つのシーケンサと１つのＣＰＵ、独立した３つのシーケンサ、独立した３つのＣＰＵで構成してもよく、また、共有される（第１～第３の手段のうち二つの手段が共有される）シーケンサと一つのシーケンサあるいは一つのＣＰＵ、共有されるＣＰＵと一つのシーケンサあるいは一つのＣＰＵで構成してもよい。図９は、図１において三つのＣＰＵのうち一つのＣＰＵを一つのシーケンサに置き換えた場合を示すシステムの概念ブロック図である。

【００７３】

【発明の効果】本発明によれば、汎用ＣＰＵを用いてボトルネックになる資源で制約される速度までヘッダ処理が低コストで高速化できる。また、シーケンサを用いて、パケットヘッダ処理をプログラマブルなハードウエア論理回路で実現することで、高速性を保ちながら、設計変更や将来の機能追加を低コストで実現できる。さらに、複数の処理をパイプラインとして動作させて、ヘッダ処理、例えば次のルータの宛先の検索とパケットに宛先データの挿入等のルーティング処理等を高速に実行できる。

【図面の簡単な説明】

【図１】本発明の一実施形態のシステムの概念ブロック図である。

【図２】本発明のヘッダ処理のシステムの概念ブロック図である。

16

【図３】本発明のヘッダ処理のシステムの概念ブロック図である。

【図４】本発明のヘッダ処理のシステムの概念ブロック図である。

【図５】本発明のヘッダ処理のシステムの概念ブロック図である。

【図６】本発明の一実施形態のシステムの概念ブロック図である。

【図７】本発明の一実施形態のシステムの概念ブロック図である。

【図８】本発明の一実施形態のシステム処理の概念ブロック図である。

【図９】図５において一つのＣＰＵを一つのシーケンサに置き換えた場合を示すシステムの概念ブロック図である。

【図１０】本発明の一実施形態のシステムの概念ブロック図である。

【図１１】本発明の一実施形態のシステムの概念ブロック図である。

【図１２】本発明の一実施形態のシステムの概念ブロック図である。

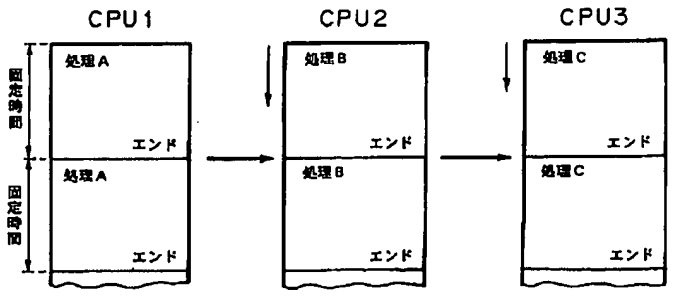
【図１３】本発明の一実施形態のシステムの概念ブロック図である。

【図１４】ＤＩＸ規定とＩＥＥＥ８０２．３規定の８０２．１Ｑ規定とのフォーマットである。

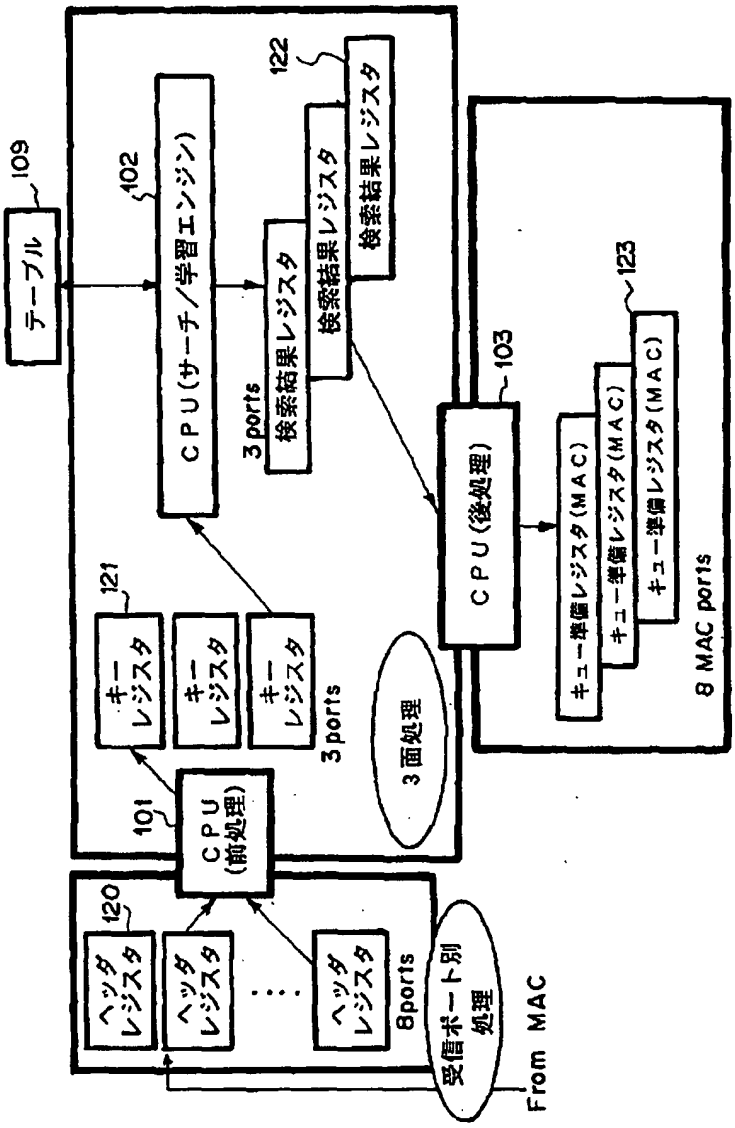
【符号の説明】

１０１　前処理シーケンサ
１０２　サーチ／学習エンジン
１０３　後処理シーケンサ
１２０　ヘッダレジスタ
１２１　検索キーレジスタ
１２２　検索結果レジスタ
１２３　キュー準備レジスタ
２０１　同期タイマ
３０１，３０２，３０３　レディレジスタ
４０１，４０２，４０３　レディレジスタ

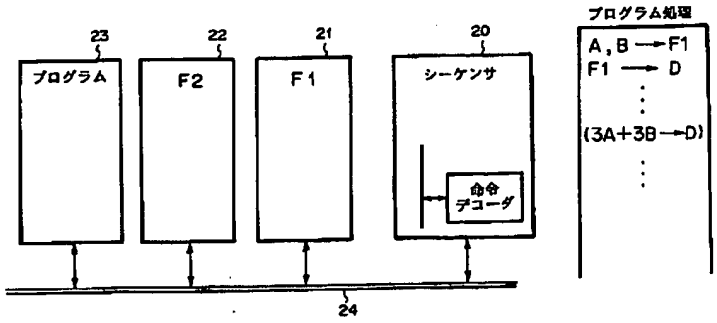
【図３】



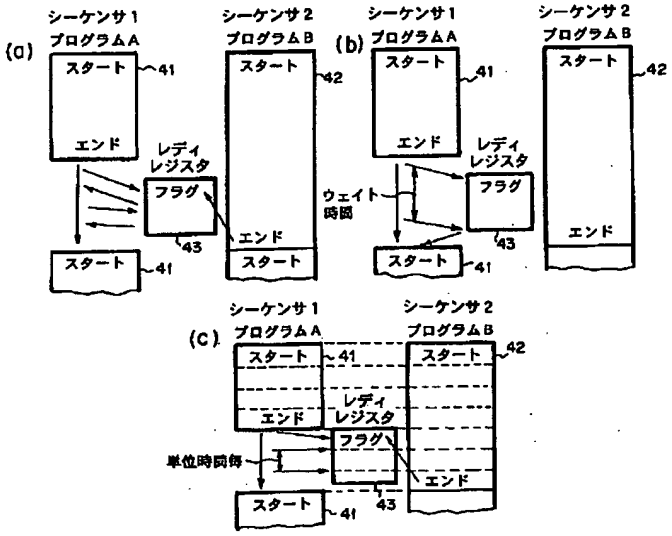
【図１】


テーブル
109
CPU(サーチ/学習エンジン)
102
3ports
検索結果レジスタ
検索結果レジスタ
検索結果レジスタ
122
CPU(後処理)
103
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
123
8 MAC ports
キー
レジスタ
キー
レジスタ
キー
レジスタ
121
3ports
3面処理
CPU
(前処理)
101
ヘッダ
レジスタ
ヘッダ
レジスタ
ヘッダ
レジスタ
120
8ports
受信ポート別
処理
From MAC

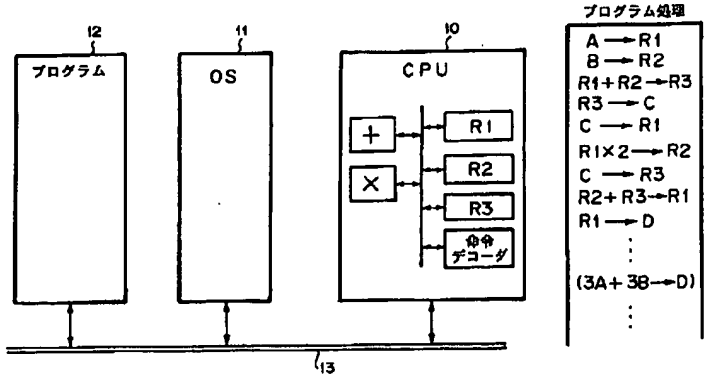
【図２】


23
プログラムA
22
F2
21
F1
20
シーケンサ
命令デコーダ
24
プログラム処理
A,B → F1
F1 → D
(3A+3B → D)

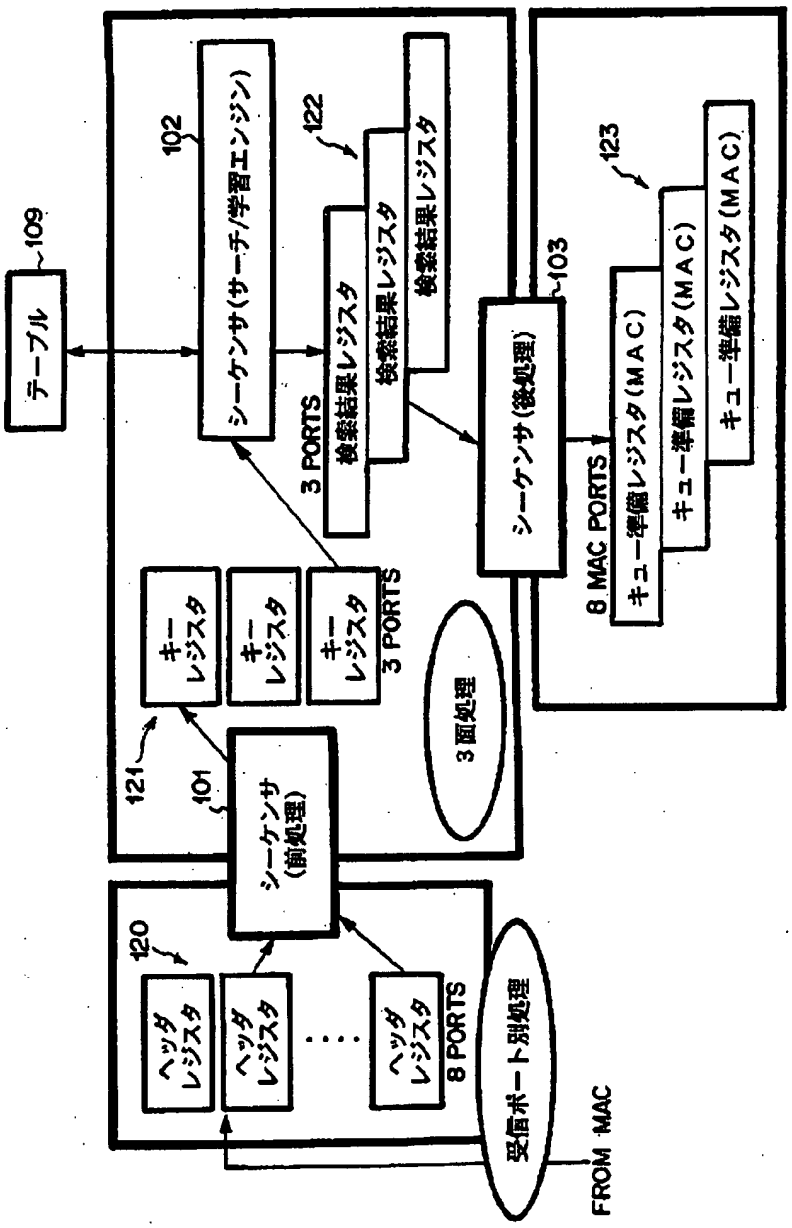
【図４】


(a)
シーケンサ1
プログラムA
スタート
41
エンド
レディレジスタ
フラグ
43
シーケンサ2
プログラムB
スタート
42
エンド
スタート
41
(b)
シーケンサ1
プログラムA
スタート
41
エンド
レディレジスタ
フラグ
43
ウェイト時間
スタート
41
シーケンサ2
プログラムB
スタート
42
エンド
(c)
シーケンサ1
プログラムA
スタート
41
エンド
レディレジスタ
フラグ
43
単位時間毎
スタート
41
シーケンサ2
プログラムB
スタート
42
エンド

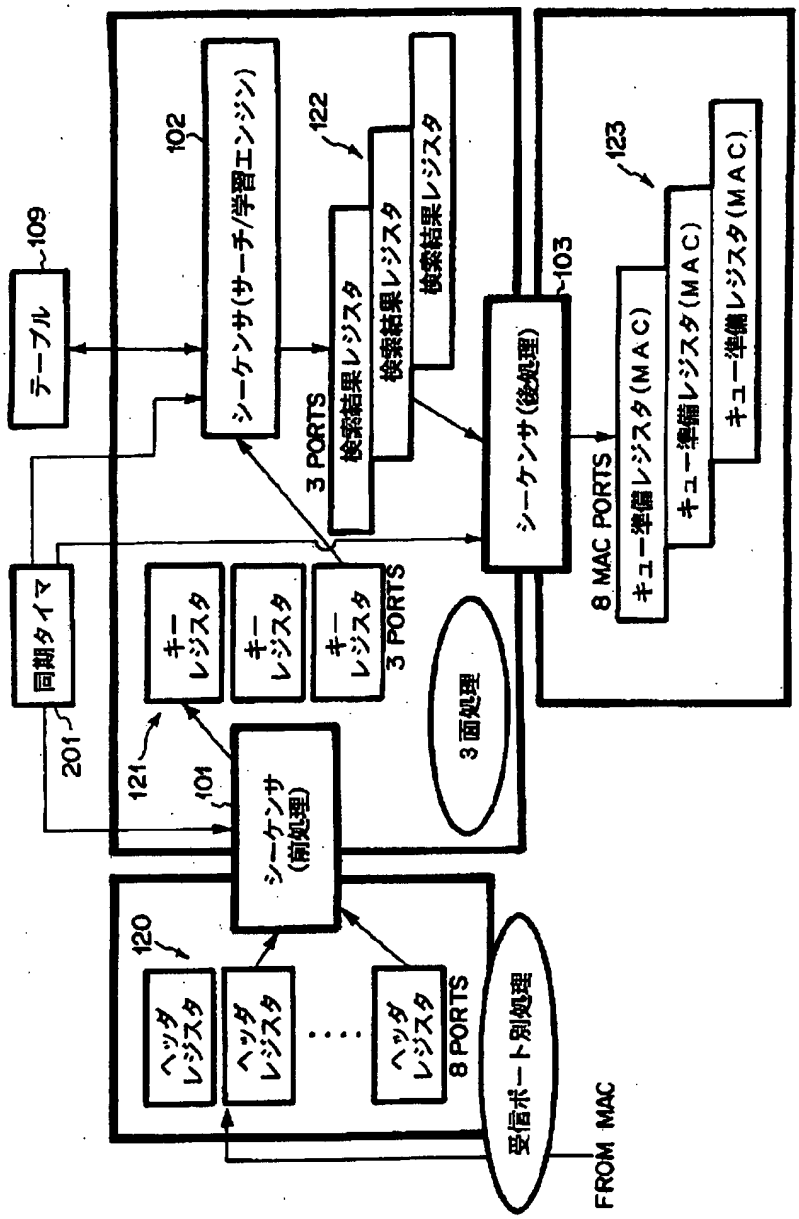
【図５】


12
プログラム
11
OS
10
CPU
+
×
R1
R2
R3
命令デコーダ
13
プログラム処理
A → R1
B → R2
R1+R2 → R3
R3 → C
C → R1
R1×2 → R2
C → R3
R2+R3 → R1
R1 → D
(3A+3B → D)

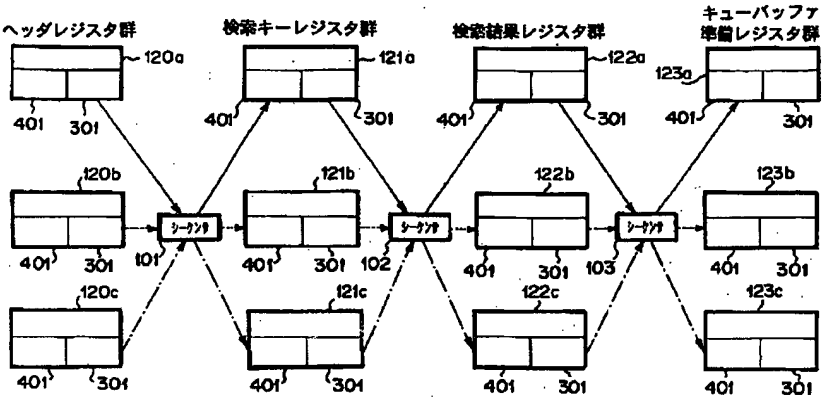
【図６】


テーブル
109
102
シーケンサ(サーチ/学習エンジン)
122
3 PORTS
検索結果レジスタ
検索結果レジスタ
検索結果レジスタ
103
シーケンサ(後処理)
123
8 MAC PORTS
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キー
レジスタ
キー
レジスタ
キー
レジスタ
3 PORTS
3面処理
121
101
シーケンサ
(前処理)
120
ヘッダ
レジスタ
ヘッダ
レジスタ
ヘッダ
レジスタ
8 PORTS
受信ポート別処理
FROM MAC

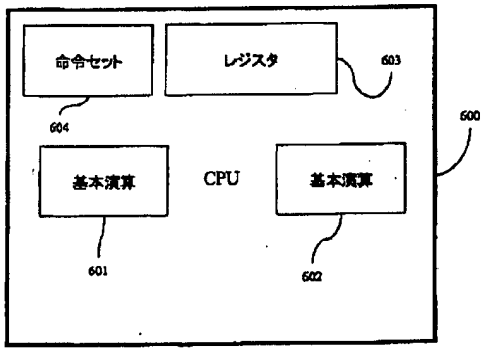
【図７】


テーブル
109
102
シーケンサ(サーチ/学習エンジン)
122
検索結果レジスタ
検索結果レジスタ
検索結果レジスタ
3 PORTS
123
103
シーケンサ(後処理)
8 MAC PORTS
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
同期タイマ
201
キー
レジスタ
キー
レジスタ
キー
レジスタ
3 PORTS
121
101
シーケンサ
(前処理)
3面処理
120
ヘッダ
レジスタ
ヘッダ
レジスタ
ヘッダ
レジスタ
8 PORTS
受信ポート別処理
FROM MAC

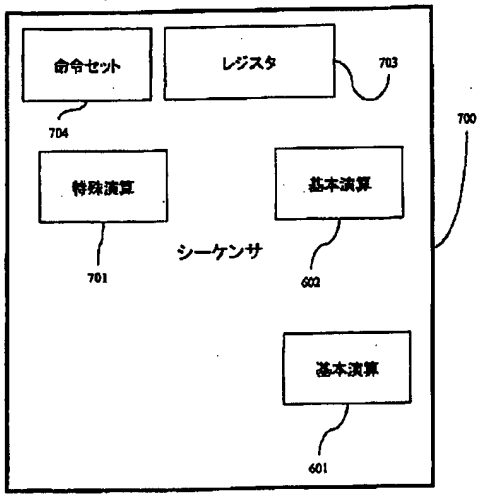
【図８】


ヘッダレジスタ群
検索キーレジスタ群
検索結果レジスタ群
キューバッファ
準備レジスタ群
120a
121a
122a
123a
120b
121b
122b
123b
120c
121c
122c
123c
401
301
101
102
103

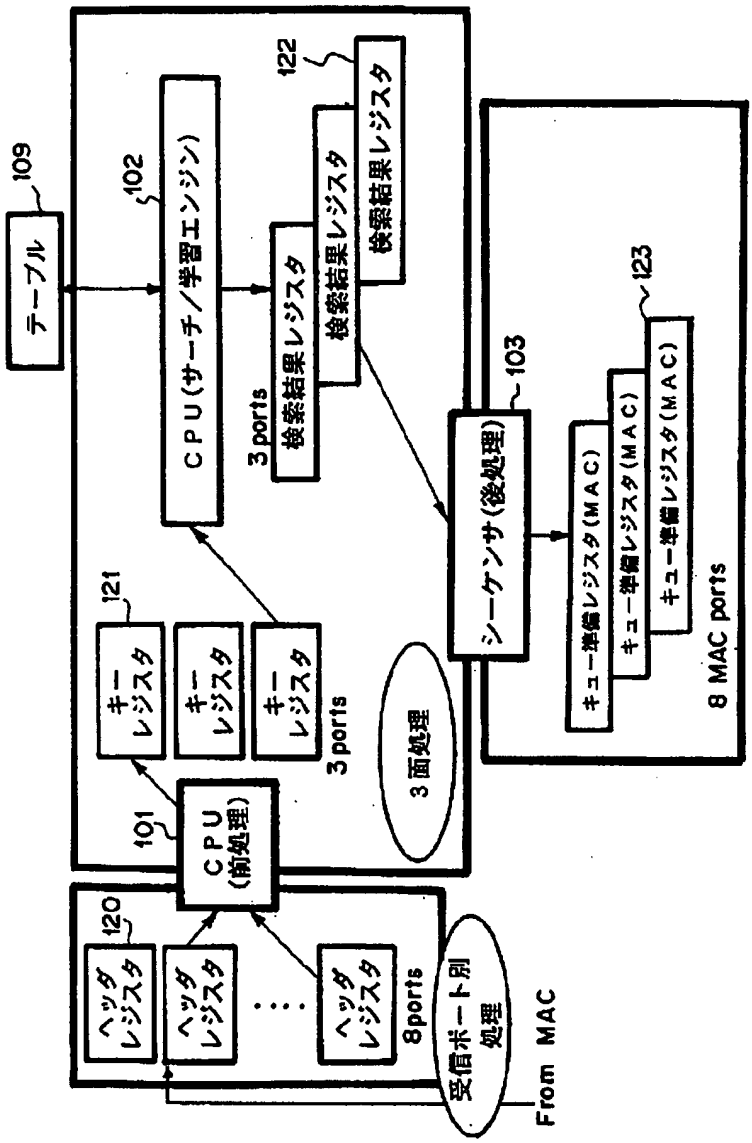
【図１１】


命令セット
レジスタ
603
604
600
基本演算
CPU
基本演算
601
602

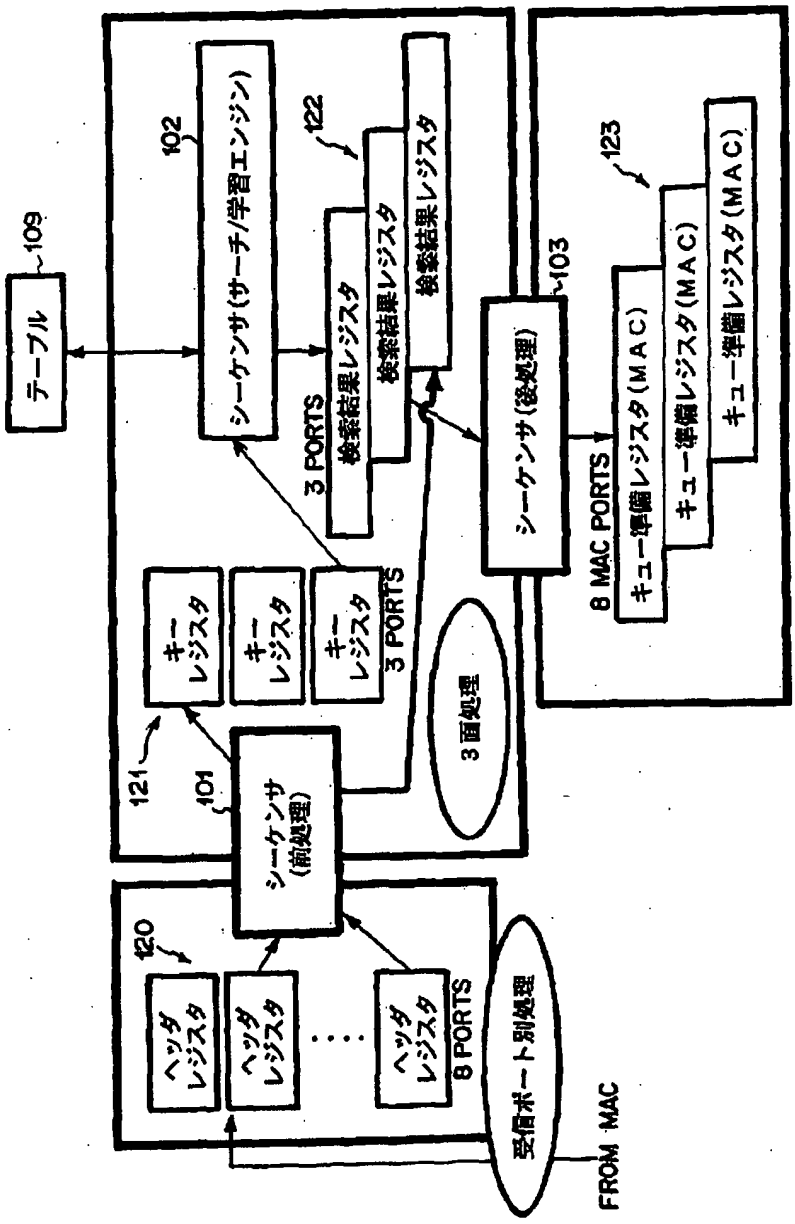
【図１２】


命令セット
レジスタ
703
704
700
特殊演算
基本演算
シーケンサ
701
602
基本演算
601

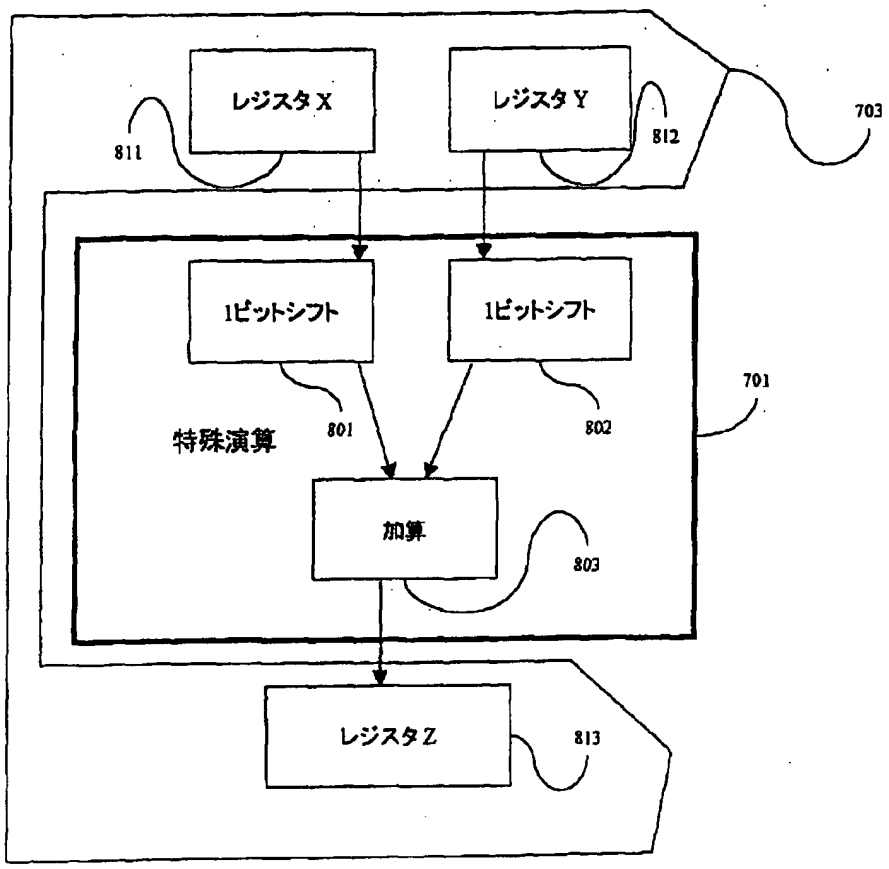
【図９】


テーブル
109
CPU(サーチ/学習エンジン)
102
122
検索結果レジスタ
検索結果レジスタ
検索結果レジスタ
3ports
121
キー
レジスタ
キー
レジスタ
キー
レジスタ
3ports
CPU
(前処理)
101
3面処理
シーケンサ(後処理)
103
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
123
8 MAC ports
ヘッダ
レジスタ
ヘッダ
レジスタ
ヘッダ
レジスタ
120
8ports
受信ポート別
処理
From MAC

【図１０】

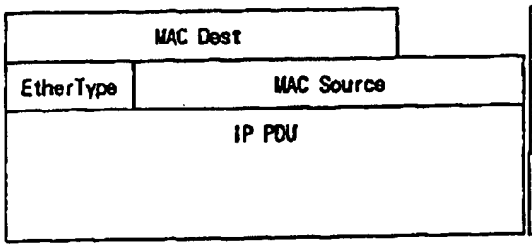

テーブル
109
102
シーケンサ(サーチ/学習エンジン)
122
検索結果レジスタ
検索結果レジスタ
検索結果レジスタ
3 PORTS
103
シーケンサ(後処理)
123
8 MAC PORTS
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キュー準備レジスタ(MAC)
キー
レジスタ
キー
レジスタ
キー
レジスタ
3 PORTS
121
101
シーケンサ
(前処理)
3面処理
120
ヘッダ
レジスタ
ヘッダ
レジスタ
ヘッダ
レジスタ
8 PORTS
受信ポート別処理
FROM MAC

【図１３】

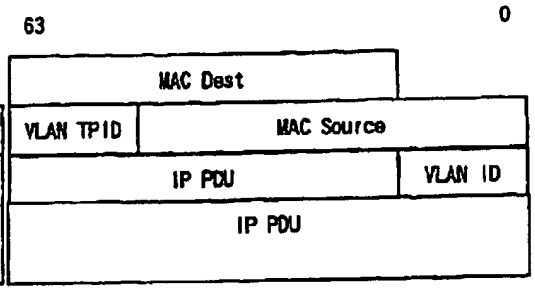

レジスタX
レジスタY
811
812
703
1ビットシフト
1ビットシフト
701
801
802
特殊演算
加算
803
レジスタZ
813

【図１４】

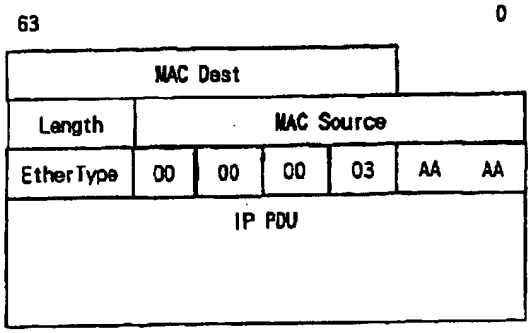
DIX Packet (non 802.1Q)



DIX Packet (with 802.1Q)



802.3 (no 802.1Q)



802.3 with 802.1Q